graphical acceleration unit 95. This embodiment includes multiple pipelines 315-319 configured to render data similar to pipelines 55-59, respectively. However, a separate computer system, referred to as master server 322, is employed to route graphical data received from client 52 to pipelines 315-319 and to control the operation of pipelines 315-319, similar to how slave control 261 of FIG. 5 controls the operation of pipelines 56-59. Other configurations may be employed without departing from the principles discussed herein. Furthermore, as previously set forth, it is not necessary to implement each pipeline 55-59 and the client 52 via a separate computer system. A single computer system may be used to implement multiple pipelines 55-59 and/or may be used to implement the client 52 and at least one pipeline 55-59.

It should be further noted that the illustrated environment has been described as utilizing X Protocol and OpenGL Protocol to render graphical data. However, other types of protocols may be utilized without departing from the principles of the illustrated environment.

Single Logical Screen Implementation

The graphical acceleration unit 95 described herein may be utilized to implement a single logical screen (SLS) graphical system, similar to the conventional system 41 shown in FIG. 2. As an example, refer to FIG. 14, which depicts an SLS graphical display system 350 in accordance with the illustrated environment. The system 350 includes a client 52 storing the graphical application 17 that produces graphical data to be rendered, as described hereinabove. Any graphical command produced by the application 17 is preferably transmitted to SLS server 356, which may be configured similarly to the conventional SLS server 45 of FIG. 2. More specifically, the SLS server 356 is configured to interface each command received from the client 52 with multiple graphical acceleration units 95a-95d similar to how conventional SLS server 45 interfaces commands received from client 42 with each graphics pipeline 36-39. The SLS server 356 may be implemented in hardware, software, or a combination thereof, and in the preferred embodiment, the SLS server 356 is implemented as a stand-alone computer workstation or is implemented via a computer workstation that is used to implement the client 52. However, there are various other configurations that may be used to implement the SLS server 356 without departing from the principles of the illustrated environment.

Each of the graphical acceleration units 95a-95d, according to the techniques described herein, is configured to render the graphical data received from SLS server 356 to a respective one of the display devices 83a-83d. Note that the configuration of each graphical acceleration unit 95a-95d may be identical to the graphical acceleration unit 95 depicted by FIG. 3 or FIG. 13, and the configuration of each display device 83a-83d may be identical to the display device 83 depicted in FIGS. 3 and 13. Moreover, an image defined by the graphical data transmitted from the application 17 may be partitioned among the display devices 83a-83d such that the display devices 83a-83d collectively display a single logical screen similar to how display devices 31-34 of FIG. 2 display a single logical screen.

To better illustrate the operation of the system 350, assume that a user would like to display an image of the 3D object 284 (FIG. 10) via the display devices 83a-83d as a single logical screen. FIG. 15 depicts how the object 284 may be displayed by display devices 83a-83d in such an example. More specifically, in FIG. 15, the display device 83a displays the top half of the object 284, and the display device 83c displays the bottom half of the object 284.

In the foregoing example, the client 52 transmits a command for displaying the object 284. The command includes the graphical data defining the object 284 and is transmitted to SLS server 356. The SLS server 356 interfaces the command with each of the graphical acceleration units 95a-95d. Since the object 284 is not to be displayed by display devices 83b and 83d, the graphical acceleration units 95b and 95d fail to render the graphical data from the command to display devices 83b and 83d. However, graphical acceleration unit 95a renders the graphical data defining the top half of the object 284 to display device 83a, and graphical acceleration unit 95c renders the graphical data defining the bottom half of the object 284 to display device 83c. In response, the display device 83a displays the top half of the object 284, and the display device 83c displays the bottom half of the object 284, as shown by FIG. 15.

Note that the graphical acceleration units 95a and 95c may render their respective data based on any of the modes of operation previously described. For example, the master pipeline 55 (FIG. 3) of the graphical acceleration unit 95a preferably receives the command for rendering the object 284 and interfaces the graphical data from the command to slave pipelines 56-59 (FIG. 3) of the graphical acceleration unit 95a. These pipelines 56-59 may operate in the optimization mode, the super-sampling mode, and/or the jitter mode, as previously described hereinabove, in rendering the graphical data defining the top half of the object 284.

In addition, the master pipeline 55 (FIG. 3) of the graphical acceleration unit 95c preferably receives the command for rendering the object 284 and interfaces the graphical data from the command to slave pipelines 56-59 (FIG. 3) of the graphical acceleration unit 95c. These pipelines 56-59 may operate in the optimization mode, the super-sampling mode, and/or the jitter mode, as previously described hereinabove, in rendering the graphical data defining the bottom half of the object 284.

Note that the master pipeline 55 (FIG. 3) of each graphical acceleration unit 95a-95d may employ bounding box techniques to optimize the operation of the system 350. In particular, the master pipeline 55 (FIG. 3) may analyze bounding box data as previously